J Netw Syst Manage (2016) 24:57–91
DOI 10.1007/s10922-014-9340-6

A Model for Incident Tickets Correlation in Network Management

Saeed Salah · Gabriel Maciá-Fernández · Jesús E. Díaz-Verdejo · Leovigildo Sánchez-Casado

Received: 26 November 2013 / Revised: 17 December 2014 / Accepted: 23 December 2014 /
Published online: 1 January 2015

Abstract In Information Technology Service Management (ITSM), network management teams typically use an Incident Ticket System (ITS) as a tool to track, troubleshoot, and coordinate the resolution of network incidents that occur during the daily operation of the network. A well organized ITS may positively impact on the efficiency of the incident management process. Nevertheless, in many cases the handling of tickets by the management team is not completely systematic and may be incoherent and inefficient. This way, irrelevant or redundant tickets for the same incident may be issued, thus creating a redundancy in the system that leads to inefficiencies. In this paper, we suggest a model aimed to correlate redundant tickets in order to reduce the information to a single ticket per incident. We validate the proposed correlation model by evaluating it with two datasets taken from a real ticketing system of a telecommunications network company. Using this model as a basis, we also develop and evaluate a methodology that assesses the efficiency of the management team during the process of tickets creation and management. Based on it, we also get some insights on the performance of the different management groups involved in the ticket creation process. These analyses can be leveraged for improving both the management groups functioning and the policies for the tickets' creation.

S. Salah (✉) · G. Maciá-Fernández · J. E. Díaz-Verdejo · L. Sánchez-Casado
Department of Signal Theory, Telematics and Communications, CITIC, University of Granada, c/Periodista Daniel Saucedo Aranda, s/n, 18071 Granada, Spain
e-mail: sasalah@science.alquds.edu

G. Maciá-Fernández
e-mail: gmacia@ugr.es

J. E. Díaz-Verdejo
e-mail: jedv@ugr.es

L. Sánchez-Casado
e-mail: sancale@ugr.es

Keywords Network management · Incident management · Incident tickets correlation · Management efficiency

1 Introduction

Over the past few years, decision makers in large management companies have realized the urgent need to enhance Service Level Agreements (SLAs) [1] with their customers, in order to increase their profits. Therefore, the use of efficient strategies for Information Technology Service Management (ITSM) is becoming one of the top demanding factors that influence the revenue of these companies. This need has emerged in parallel with the evolution, in terms of scalability and complexity, of telecommunication networks and the criticality of the services that they offer. For this reason, network management teams are always trying to find efficient methods to improve network uptime and to quickly solve network incidents to be within the committed thresholds established by SLAs [2].

One of the most important components of ITSM is the incident management, which provides mechanisms to recognize, isolate, correct, and log incidents that occur during the daily operation of the network. These incidents are considered as the most significant contributors to system downtime and may disturb the service provisioning [3, 4], *e.g.*, software bugs, hardware failures, or device misconfigurations, among others.

In this context, the Information Technology Infrastructure Library (ITIL) is the most widely adopted approach for ITSM in the world [5]. It is the best practices standard in managing information technology services that provides infrastructure, development, and operations for identifying, planning, delivering, and supporting the IT services to the business. Incident management is one of the main processes that ITIL provides. By referring to the ITIL terminology, the incident can be defined as *"an unplanned interruption to an IT service or reduction in the quality of an IT service. Failure of a configuration item that has not yet impacted service is also an incident"* [6].

Incident Ticket Systems (ITSs), or Service Desks as mentioned in ITIL, have been introduced as the main tool to assist in the incident management process. They are defined as databases that share reports in a specific form-based structure, and actions to create, update, and resolve any network incident reported by customers, organization employees, or monitoring systems. They might also contain administrative information about customers, workarounds to be applied for common incidents, and other similar data.

Normally, there are two main different procedures for creating incident tickets. On the one hand, they can be created by the management team in response to a network failure discovered by management software, *e.g.*, alerts from OpenView [7] or other network management platforms. On the other hand, they can also be created by the customer care staff in response to a call from an end user facing problems in accessing some services or applications. Incident resolution begins with the creation of a ticket that contains the information describing the ongoing incident. Next, the

ticket may be sequentially reassigned to several technical groups involved in the incident solving procedure. Finally, it is closed when the incident is completely solved and, optionally, a crosscheck of the solution can be made. Therefore, the ticket may pass through several hands and undergo various degrees of escalation with respect to incident severity or customer priority before being completely acknowledged.

Although ITS plays an important role in the incident management, the process of creating the tickets is not completely systematic and may be incoherent and inefficient. For example, the management team may create so-called irrelevant tickets, i.e., tickets that do not reflect the existence of an actual network problem; or it may create multiple tickets that are related to the same incident. It is also possible that different staff groups or departments in a company create different tickets for a single incident.

In this context, we present three main contributions in this paper. First, a novel model for an incident ticket is proposed. Unlike others, we try to make the model as general as possible by focusing only on the generic fields that appear in tickets such as staff information, temporal and topological information, etc. Second, we propose an incident ticket correlation model targeted at improving the ticketing system to achieve the ideal situation at which just a single ticket per incident exists. Third, we propose some metrics used to measure the redundancy in the ITS, according to the proposed correlation method. These metrics are then used to evaluate the procedures for creating tickets.

To the best of our knowledge, this is a novel approach, as there are no systematic methods currently available for this purpose. Previous work in ITSs has focused on improving the whole system to achieve the corresponding SLA in time. Notwithstanding with this, few efforts have been devoted to optimize the efficiency of the ticket creation process itself, which, at the same time, would improve the overall task of solving the incidents. In our opinion, this is partly due to the fact that the ticketing information is confidential, and this fact hides both the real tickets and the used procedures from the research community. Furthermore, vendors are not interested in publishing the data nor the procedures due to competitive issues, while management teams either do not keep track of this data or treat it as confidential information due to similar motives. In this work, we focus on dealing with the ticket creation process itself.

The remainder of this paper is organized as follows. Section 2 provides a brief survey of related work. The incident ticket lifecycle is presented in Sect. 3, while a model for the incident tickets is proposed and discussed in Sect. 4. Next, this model is used as the basis for a correlation process described in Sect. 5. The correlation method is experimentally evaluated in Sect. 6. Section 7 describes, through a case study, the application of the method for the evaluation of the performance of the actors involved in ticket creation. Finally, Sect. 8 presents some conclusions and provides some insights about further work.

2 Related Work

A thorough review of the literature of ITSs and correlation shows several research tendencies, as emphasized by Lewis and Dreo [8]. In this survey, the authors analyzed the challenges and research trends for extending ITSs for the automatic generation of tickets, the diagnosis of faults and the correlation of multiple views of network incidents and behavior. For this, Lewis and Dreo suggested various techniques, as the use of filtering and grouping of tickets with respect to language, function, time, and topology; the use of Rule-based Reasoning (RBR) or Case-based Reasoning (CBR) for acquisition and representation of fault diagnostics knowledge; or the use of Fuzzy Logic (FL) for correlating multiple views of network incidents.

A similar work was done by Johnson [9], who proposed the RFC 1297. This report mainly discussed the most important extensions that can improve the network operations efficiency, and emphasized the importance of ITSs for network management teams. Nevertheless, it did not provide a methodology to analyze the content of the ticket itself.

Dreo [10] went forward and proposed the use of ticket correlation for the discovery of problems and access to problem-solving expertise. One of the main conclusions of Dreo's work is that a high-quality ticket correlation needs to use good models for the functional and topological aspects of any network service. For this reason, the authors use the topological and temporal information as the main dimensions for the correlation process.

Miao et al. [11, 12] focused on enhancing the ticket management lifecycle by proposing a unified ticket generative model that characterizes the lifecycle of a ticket using both the content and the routing sequence of the ticket and a Markov model-based approach to predict the resolver of the ticket based on the expert group that previously processed the ticket. The aim is to enhance the routing and minimize the number of transfer steps before it reaches a resolver.

Tang et al. [13] suggested an automatic approach to discover the false negatives from those incident tickets that are created by humans in order to improve the configurations of the monitoring system. They applied a text classification model for analyzing the descriptions of incident tickets and identifying the corresponding system issues.

The work done by Li et al. [14] is intended to give an estimate of the mean effort for incident tickets, by means of a two-stage approach. First, a meta-model is proposed, and some handling priority rules are used to compute what the authors call "attention duration". Then, a Maximum Likelihood (ML) approach is used to estimate the mean effort for a ticket class by using such attention information.

However, in spite of the above works about ticket management, we did not find a lot of work focusing on the ticket correlation process itself in the literature, except for some preliminary works that were done in order to achieve different purposes. Next, we summarize these efforts.

Some authors tried to correlate information from ITSs with another source of information, as in [15–18]. In these papers, the authors proposed models to relate two types of data sources, the so-called source tickets, which are created by the monitoring system, and the service tickets, which are generated by the management

team. Their main goal was trying to improve the accuracy and effectiveness of the management process in real time. The authors concluded that ticket overlapping is one of the main challenges for their models, but they did not provide any solutions to handle it.

A similar work was also proposed in [19–22], in which the authors used simple ticket preprocessing operations to reduce the total number of tickets before correlating them. Nevertheless, the focus of these papers was to study and characterize the nature and causes of routing changes and the observed instability. Therefore, they did not deeply analyze the ITSs, and the correlation of the tickets is not targeted at reaching one ticket per incident.

Other approaches used different techniques to process the tickets and extract the description of the incidents, like Potharaju et al. [23]. In this work, the authors proposed NetSieve, a system that analyzes natural language text in network tickets to infer the problem symptoms, troubleshooting activities, and resolution actions. They used statistical Natural Language Processing (NLP), knowledge representation, and ontology modeling to achieve these goals. Others, like Medem et al. [24] used machine learning techniques to process the tickets and extract a concise description of the incidents. To do that, they modeled each ticket as a vector of keywords whose frequencies are used as weights to create a hierarchical clustering. The resulting clusters are used to correlate the tickets and extract the incident. However, despite the usefulness of these techniques of relating tickets that share the same keywords, it was limited to just work on some free-text fields and ignore some important features such as timestamps and topological information.

In addition to the papers described above, some researchers used ITSs for other purposes. For example, Tanaka [25] and Pándi [26] made some statistical measurements and characterized tickets from several networks taken as case studies. From these analyses, the authors proposed some recommendations to enhance the performance of ITSs and their use in the context of SLAs. They also presented a comprehensive and detailed taxonomy for the categorization of equipment, services, and users affected by events. However, in their work they did not focus in preprocessing operations or the correlation of tickets from the point of view of reducing the number of tickets per incident.

Other authors used ITSs as an assessment tool to validate their research ideas. This is the case of the work done by Huang et al. [27]. They proposed to use network-wide analysis of routing information to diagnose network disruptions. In a similar work, Labovitz et al. [28] made an experimental study of Internet stability and the origins of failure in Internet backbones utilizing the information provided by the ITS.

Finally, despite the availability of many commercial service desk products, such as HP Service Center [7], ServiceNow [29], BMC Remedy [30], or Tivoli SCCD [31], few efforts have been devoted to optimize the ticket creation process itself. Most of these tools use their own procedures for resolving network incidents and enhancing the whole system to recover the SLA in time. For the time being, and to the best knowledge of the authors, the procedures used for incident tickets correlation are not publically available, because vendors of these tools are not interested in publishing the procedures and keep them hidden from the public due to

competitive issues. So, the inherited complexity of these tools makes it difficult to inspect which type of correlation operation they use, if any.

3 The Incident Ticket Lifecycle

As previously discussed, our principal motivation is to find mechanisms for building a simple and manageable system to deal with incidents in a network. ITSs are a main tool used to record and report incidents within an operational system. They are, therefore, considered as a main tool for tracking resolution activities associated with incidents. For this reason, our work here will focus on the management of tickets, under the hypothesis that they are the best tool for representing an incident lifecycle. Nevertheless, it is worth to mentioning here, that although there is a direct relationship between tickets and incident lifecycles, there could be significant differences between them that will be discussed in what follows.

For these reasons, our starting point is the understanding of such a ticket lifecycle. In Fig. 1 we show the different stages and actions involved in a generic incident ticket lifecycle: *ticket creation, ticket assignment, ticket management, ticket reassignment, ticket resolution,* and *ticket validation.*

- ***Ticket creation:*** A ticket is created by a network management team or administrative staff, here denoted as *ticket creator*, either (i) in response to the reception of network alerts triggered by proactive network monitoring systems due to service disruptions or failures of network elements, or (ii) in response to a call or notifications typically coming from customer care units or directly from end users who face technical problems in the access to services or applications.

In a normal situation, the time elapsed between the origin of an incident and its corresponding ticket creation time should not be long, although in many situations it

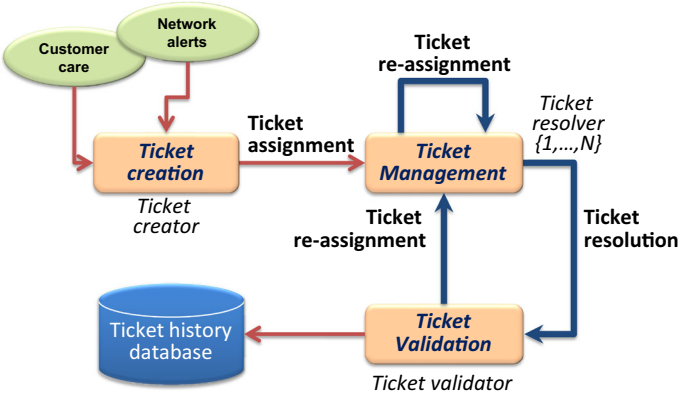

Fig. 1 Generic incident ticket lifecycle

can be in the range of minutes or hours, especially when tickets are created manually and not by an automated system. In these situations, the ticket creator may register the timestamp corresponding to the incident as reported by the monitoring systems, or the time when a customer reported a failure or a management team member detected it.

- ***Ticket assignment:*** After being created, a ticket is then handed off to an appropriate technical person or group in the company, denoted as *ticket resolver*, who is expected to resolve the incident.
- ***Ticket management:*** This is the main stage in the ticket lifecycle. During it, the ticket resolver analyzes the incident reported in the ticket, finds potential solutions, tests them, and reports the results.
- ***Ticket reassignment:*** It could happen that the ticket resolver, after having investigated the problem–and possibly having also applied several solutions—detects that the incident solution is out of the scope of his/her responsibilities. In this situation, the ticket resolver will again hand off the ticket to another group for its management. This process is the ticket reassignment. In general, when this happens, a ticket can be managed by N different ticket resolvers, and therefore, can go through the ticket management stage several times.

Normally, before doing a reassignment, the ticket resolver registers (in the ticket itself) the different activities carried out to solve the incident and his/her results.

- ***Ticket resolution:*** Once the incident reported on the ticket is solved or a workaround discovered, the ticket resolver registers the final solution and the corresponding tests and results applied to the ticket. Finally, he/she notifies that the incident is resolved.
- ***Ticket validation***: After the ticket resolution, the ticket arrives to another member or group, denoted as *ticket validator* that is in charge of verifying whether the solution is satisfactory or not. If it is validated, the ticket is saved in a *ticket history database* that is commonly used to build knowledge for future incidents. On the contrary, if the ticket solution is not validated, the ticket is reassigned again to a ticket resolver. Therefore, the *ticket lifetime* can be defined as the time elapsed between the creation of a ticket and its resolution or validation in some cases.

4 A Novel Model for Incident Tickets

ITSs produce tickets in different and heterogeneous formats, containing a wide variety of fields with several possible values for them. Some of these fields are inherent to the operation of the ITS, and therefore, they could be considered as

generic fields. Others are specific, i.e., they collect specific requirements established by the different incident analysts.

Here, we make an effort to define a generic model for a ticket. Thus, we focus on the generic fields that appear in tickets and assume that every ticket will contain these common fields. The proposed model is based on two categories of common fields. In what follows, we classify and describe them:

- ***Timestamps fields:*** This category contains the most important time indexes that are mainly related to the lifecycle of the tickets.

  Ticket creation time for ticket i (t_i^{CT}): It is the instant at which the ticket creator generates the ticket in the ITS.
  Ticket resolution time for ticket i (t_i^{RT}): This time is recorded on the ticket when the solution for the ticket is notified by the ticket resolver to the ticket validator.
  Ticket validation time for ticket i (t_i^{VT}): It represents the instant at which the ticket validator acknowledges the solution given to the incident, sending the ticket to the ticket history database.

  For every ticket i, we define a list of timestamps T_i^{TS} as:

$$T_i^{TS} = \left\{t_i^{CT}, t_i^{RT}, t_i^{VT}\right\}$$

- ***Identifiers' fields***: This category of fields in a ticket is used to univocally identify the ticket in the database, associate it to the different ticket resolvers involved in the ticket management stage, and also relate it with the network elements or services affected by the incident reported in the ticket.

  For each ticket i, we define three types of field lists:

  Ticket IDs (*TIDs*): Each ticket in the database has a unique identifier that is typically an alphanumeric value representing the unique reference of the ticket itself. We will refer to this identifier as the *main ticket ID.*

Although there is a unique *main ticket ID* field in a ticket, during the management stage, ticket creators/resolvers normally may directly or indirectly mention the main ticket IDs of other tickets. This reference is usually included in other fields, especially those with free text format, such as incident description, worklogs or incident resolution fields. For this reason, we also consider a list of *ticket IDs* extracted from these fields and containing all these related identifiers. For every ticket *i,* we have

$$T_i^{TID} = \left\{TID_i^1, TID_i^2, \ldots, TID_i^n\right\}$$

where TID_i^1 is the main ticket ID for ticket i.

Group IDs (*GIDs*): Due to the fact that every ticket can be managed by different ticket resolvers during its lifecycle, the tickets contain information about them.

Here, we define a list of ticket resolvers (*TR*) involved in the management of a ticket. Every ticket resolver j that was involved in the management of ticket i is identified by TR_i^j, resulting in the list

$$T_i^{TR} = \{TR_i^1, TR_i^2, \ldots, TR_i^m\}$$

where TR_i^1 is the identifier for the first ticket resolver of the ticket, that is, the ticket resolver chosen by the ticket creator in the assignment action for the ticket i.

Object IDs (*OIDs*): Each node, service, component, or element in a network is usually referred to with a unique identifier, usually a string, that we call the *object ID*. When an incident is detected in a network, the corresponding ticket is associated with the identifier of an object of the network that we call the *main object ID*. In addition, in the ticket creation process or along the ticket management stage, the ticket creators/resolvers usually include other *object IDs*, mainly in the incident description, worklog, or incident resolution fields, which are related to the incident described by the ticket.

Thus, we define a list of *OIDs* for every ticket i, that contains all the previously described objects' identifiers.

$$T_i^{OID} = \{OID_i^1, OID_i^2, \ldots, OID_i^l\}$$

where OID_i^1 is the main object ID for ticket i.

In summary, our model represents a ticket by a tuple containing the following elements (note that every element is really a list of values):

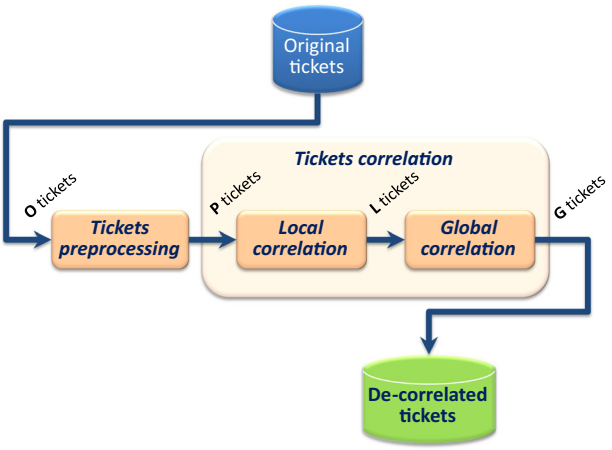

Fig. 2 Proposed incident ticket correlation process

$$T_i = T_i^{TS}, T_i^{TID}, T_i^{TR}, T_i^{OID} \quad (1)$$

It should be pointed out that tickets normally include many other fields not considered in our model, *e.g.*, priority of the ticket. We have only included those fields that will be used for the correlation algorithms proposed here.

5 Incident Ticket Correlation Process

In this section we propose a ticket correlation process aimed to enhance the information provided by the ITS, trying to rebuild the ITS database in order to obtain, for every incident in the network, only one ticket containing accurate information about that incident.

The global procedure that we propose can be decomposed in two main phases: *tickets preprocessing* and *tickets correlation* (Fig. 2). Thus, starting from a set $\mathcal{O}$ of O original tickets stored in the ITS, a first preprocessing step is required in order to extract only meaningful tickets, as will be explained next. The preprocessing phase is divided into two subphases: *selection of incident-related tickets* and *tickets parameterization*. The resulting set, $\mathcal{P}$, composed of P tickets, is then passed through the ticket correlation procedure to produce a new set, $\mathcal{G}$, composed by G supposedly de-correlated tickets (that is, the final set of tickets) that is expected to contain a single ticket per incident. As shown in Fig. 2, the proposed correlation procedure can be further split into two subphases: *local correlation* and *global correlation*, each of them targeted at removing the redundant tickets according to different properties. In the following subsections, we provide a more detailed discussion about each step.

5.1 Tickets Preprocessing

The first step for dealing with the original ITS database, $\mathcal{O}$, is to apply a preprocessing mechanism aimed to obtain, in a proper format, the tickets that are relevant from the point of view of incident solving. To do this, we divide the preprocessing phase into two sub-phases:

5.1.1 Selection of Incident-Related Tickets

The first task is selecting only those tickets that are related to incidents. Note that there are many tickets in the ITS that have not really been created because of an incident occurrence. First, we can find so called *malformed or void tickets*: those created for doing nothing either due to a failure in the network management tool, which automatically generates a large number of such kind of tickets, or due to misbehaviors of the management staff, they sometimes create many tickets without a real network incident. This process depends on the specific input dataset.

Typically, a ticket is not considered as incident-related if it does not contain a number of required fields filled with data, mainly those describing the affected

nodes, services, and the resolution time. Any ticket that does not contain such kind of information is considered malformed and should be removed from the whole procedure. Second, we can also find so called *irrelevant tickets*: those which are not related to real network incidents. In many ITSs it is usual that they include tickets for topics not directly related to network incidents but contain information about some other issues as, for example, the availability of a new software release in certain network nodes or maintenance activities. These irrelevant tickets are mainly not related to network incidents, and therefore, should be removed, according to the target of the whole procedure. To identify these tickets, any matching procedure could be followed. In our case, we identify those tickets that contain any word from a list of keywords built with the help of the management staff as irrelevant.

5.1.2 Tickets Parameterization

In this step, and after having selected incident-related tickets, we extract the list of parameters of the proposed ticket model, as illustrated in Eq. (1). To do this, for every ticket we extract these relevant fields as follows: *main object ID* (OID_i^1), *main ticket ID* (TID_i^1), *first ticket resolver ID* (TR_i^1), and the list of timestamps $\left(T_i^{TS}\right)$ are normally mapped in a ticket field, so they are extracted directly from their corresponding fields in the tickets. The other identifers in the model are extracted by using pattern matching methods to search for the occurence of keywords and text conformation to given formats in other tickets' fields; normally in incident description, worklogs, and solution fields.

After applying the above two steps, the output of the preprocessing step is a set $\mathcal{P}$ of P tickets, T_i, in the standard form given in Eq. (1):

$$\mathcal{P} = \{T_1, T_2, \ldots, T_P\}$$

5.2 Tickets Correlation

The tickets' correlation phase has two main goals. First, it is used to reduce the total number of tickets by substituting every subset of tickets that share some common properties into a single one that is termed as the *representative ticket* for that subset. Second, the correlation process is expected to increase the semantic information of the data of the ticket. This means that a representative ticket will give more accurate information about the real incident than the bunch of tickets taken as input to this phase. This is mainly because the information provided by them summarizes all the information in the tickets related to the same incident.

We divide the ticket correlation phase into two subphases: local and global correlations. In the following we provide a detailed description of each one of them.

5.2.1 Local Correlation

Local correlation is used to correlate tickets that collect information about different problems reported for the same network node or service but really correspond to the same incident. Our hypothesis here is that if several tickets have the same *main*

object ID (same that node) and they overlap in time, those tickets are related to the same incident, and thus, should be correlated. We call this process a local correlation process because it works on a per-node level and ignores any tickets potentially related to the same incident but reported for different main object IDs. These representative tickets will be treated as input to the next subphase, global correlation.

For the local correlation process, we propose an algorithm that takes several overlapped tickets having the same main object ID and reduce them to a single ticket, T'_R, termed as the *local representative ticket*. The algorithm is explained in what follows.

Suppose that, from the complete tickets database, we identify the subsets $S_k \subset \mathcal{P}$, each composed by a number N_k of tickets $S_k = \{T_1, T_2, \ldots, T_{N_k}\}$, so that $\cup_{\forall k} S_k = \mathcal{P}$ and $S_i \cap S_j = \emptyset, \quad \forall i \neq j$. Each subset S_k is composed of several tickets in $\mathcal{P}$ that share the following properties:

(i) They have the same main object ID, i.e.,

$$OID_i^1 = OID_j^1, \quad \forall i, j \in [1, N_k]$$

(ii) They overlap in time; that is, for every ticket $T_i \in S_k$, we can find at least another ticket $T_j \in S_k$ so that their intervals $\left[t_i^{CT}, t_i^{RT}\right]$ and $\left[t_j^{CT}, t_j^{RT}\right]$ overlap, i.e.,

$$\left\{\left\{t_i^{CT} \leq t_j^{CT} \leq t_i^{RT}\right\} \quad OR \quad \left\{t_j^{CT} \leq t_i^{CT} \leq t_j^{RT}\right\}\right\} = true$$

(iii) There are no time gaps between them, i.e.,

$$\forall t \in \left\{ \min_{i \in [1,N_k]} \left\{t_i^{CT}\right\}, \max_{i \in [1,N_k]} \left\{t_i^{RT}\right\} \right\}, \quad \textit{there is at least one active ticket}$$

For every subset S_k, the correlation algorithm creates a *local representative ticket*, T'_K, which will replace all the tickets in S_k, following these rules:

$$t_k'^{CT} = \min_{i \in [1,N_k]} \left\{t_i^{CT}\right\} \tag{2}$$

$$t_k'^{RT} = \max_{i \in [1,N_k]} \left\{t_i^{RT}\right\} \tag{3}$$

$$t_k'^{VT} = \max_{i \in [1,N_k]} \left\{t_i^{VT}\right\} \tag{4}$$

$$T_k'^{x} = \bigcup_{i=1}^{N_k} T_i^x, \quad \text{where } x = \{TID, TR, OID\} \tag{5}$$

$$OID'^{1}_{k} = OID^{1}_{i}, \quad \text{for any } i \tag{6}$$

Note that the value for the fields T'^{TID}_{k}, T'^{TR}_{k}, and T'^{OID}_{k} is extracted as the union (concatenation without repetition) of the corresponding lists in the tickets from the subset S_k. This means that, in the representative ticket, the meaning of TID'^{1}_{i} and TR'^{1}_{i} as the main ticket and ticket resolver IDs is lost. However, the value OID'^{1}_{i} still represents the main object ID, as this value is the same for every ticket in S_k.

It must also be noted that it could happen that a subset S_k is composed of a single ticket, in case that this ticket is not overlapped with any other in $\mathcal{P}$ for the same OID[1]. In such a case, this ticket is itself the representative ticket for that subset.

The output of this local correlation process is a set $\mathcal{L}$ of L representative tickets, each one representing a different subset $S_k \subset \mathcal{P}$.

$$\mathcal{L} = \{T'_1, T'_2, \ldots, T'_L\}$$

The function of the algorithm can be explained through a simplified example as shown in Fig. 3. In this example we consider one subset, S_k, composed of three tickets ($N_k = 3$). As depicted in Fig. 3a, the creation time of the ticket T_2 $\left(t^{CT}_2\right)$ occurs during the interval $[t^{CT}_1, t^{RT}_1]$; besides, T_3 is created (t^{CT}_3) within the interval $[t^{CT}_2, t^{RT}_2]$; finally, these three tickets have the same main object ID (*e.g.*, node *x*), *i.e.*, $OID^1_1 = OID^1_2 = OID^1_3 = x$. Thus, T_1, T_2, and T_3, according to the proposed method, should be locally correlated into the *representative ticket* (T'_r), as shown in Fig. 3b, where $t'^{CT}_r = \min_{i\in[1,3]}\{t^{CT}_i\} = t^{CT}_1$ and $t'^{RT}_r = \max_{i\in[1,3]}\{t^{RT}_i\} = t^{RT}_3$. Besides, it must be noted that T'^{TID}_r, T'^{TR}_r, and T'^{OID}_r are extracted as the union of the corresponding lists in T_1, T_2, and T_3.

5.2.2 Global Correlation

After having correlated the overlapping tickets related to the same main object ID (local correlation), here we extend our correlation process to consider other tickets

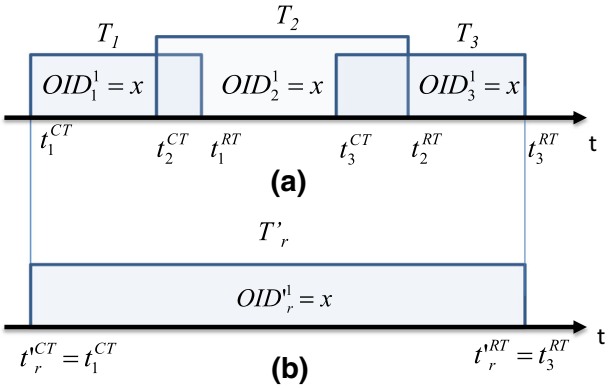

Fig. 3 Local tickets correlation example: **a** original tickets, **b** representative ticket

with different main object IDs, leading to a global correlation. The aim now is to identify tickets' reporting problems in different nodes but corresponding to the same incident.

In the ticket creation process or along the resolution of the incident, ticket creators/resolvers usually include information in a ticket about other tickets (*TIDs*) or other objects in the network (*OIDs*) that are related to the incident described by the ticket. This information appears, for a ticket i, in the lists T_i^{TID} and T_i^{OID}.

In a first approach, we could say that two tickets i and j are related to the same incident if one of the following conditions is fulfilled

$$\text{Condition 1} : \exists TID/[TID \in T_i^{TID} \text{ and } TID \in T_j^{TID}]$$
$$\text{Condition 2} : \exists OID/[OID \in T_i^{OID} \text{ and } OID \in T_j^{OID}]$$

Note that although we impose this restriction, the simple existence of a common *TID/OID* value in two tickets does not necessarily imply a relationship between them. Indeed, it is common that a ticket includes a reference to another *TID/OID* where a workaround for solving the incident can be found. For this reason, we impose an additional constraint for relating two tickets to the same incident: temporal overlapping.

Finally, we make a final assumption. We assume that the transitive property is valid in the relationships among tickets, *i.e.,* if tickets i and j are related, and tickets j and k are also related, then tickets i, j, and k are all related.

The global correlation has the same target as the local correlation, that is, to obtain only one ticket per incident; however, as a difference in this case, it will be referred to different nodes. Therefore, it is considered as an almost identical process as the local correlation, in which every subset of related tickets in $\mathcal{L}$ is substituted by a single *global representative ticket,* denoted as $\bar{T}$. The complete algorithm is described in what follows, discussing only the main differences with the local correlation process.

Suppose that, from the set $\mathcal{L}$, we identify different subsets $C_k \subset \mathcal{L}$, such that $\cup_{\forall k} C_k = \mathcal{L}$ and $C_i \cap C_j = \emptyset, \quad \forall i \neq j$. Each subset C_k is composed of a number N'_k

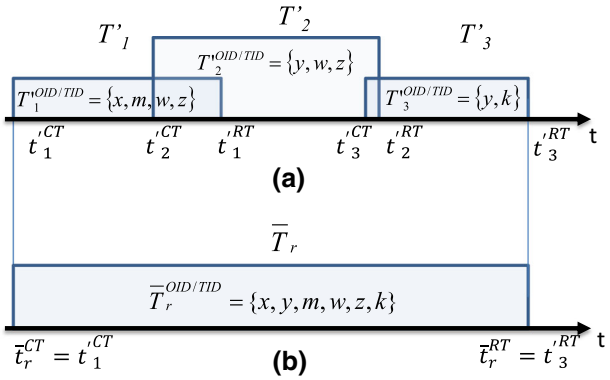

Fig. 4 Global tickets correlation example: **a** original tickets, **b** representative ticket

of tickets $C_k = \left\{T'_1, T'_2, \ldots, T'_{N_k}\right\}$, namely all the tickets contained in C_k share the last two properties as in the local correlation, *i.e.,* properties (ii) and (iii). Here, the first property is defined as follows: Every ticket $T'_i \subset C_k$ is related to at least one ticket $T'_j \subset C_k$ if $\left\{\left\{T_i'^{TID} \cap T_j'^{TID}\right\} OR \left\{T_i'^{OID} \cap TT_j'^{OID}\right\}\right\} \neq \emptyset, \quad \forall i,j \in [1, N_k]$.

The global correlation process creates a *representative ticket*, $\bar{T}_k$, for the subset of tickets C_k, which will replace all the tickets in C_k, according to the same rules presented in the local correlation section, apart from that T' is replaced by $\bar{T}$ and t' is replaced by $\bar{t}$.

It is important to mention here that, the values for the fields $\bar{T}_k^{TID}$, $\bar{T}_k^{TR}$, and $\bar{T}_k^{OID}$ are extracted as the union of the corresponding fields in the tickets from the subset C_k. After this process, the meaning of OID'^1_i, TID'^1_i, and TR'^1_i as the main object, ticket and ticket resolver IDs are lost.

Figure 4 shows an example of the application of the algorithm using three local representative tickets ($N'_k = 3$). As depicted in Fig. 4a, $T_1'^{OID} = \{x, m, w, z\}$, $T_2'^{OID} = \{y, w, z\}$, and finally $T_3'^{OID} = \{y, k\}$. T'_1 and T'_2 are related, since they have two OIDs in common, namely w and z, and they temporally overlap. Furthermore, T'_2 and T'_3 are also related since they have one OID in common, namely y, and they temporally overlap. Therefore, the three local representative tickets are correlated into a global representative ticket $\bar{T}_r$ as illustrated in Fig. 4b having $\bar{t}_r^{CT} = \min_{i\in[1,3]}\left\{t_i'^{CT}\right\} = t_1'^{CT}$, $\bar{t}_r^{RT} = \max_{i\in[1,3]}\left\{t_i'^{RT}\right\} = t_1'^{RT}$ and finally $\bar{T}_r^{OID/TID} = \{x, y, m, w, z, k\}$.

The output of this global correlation is a set $\mathcal{G}$ of G tickets, which is composed of the list of representative tickets, each one for every different subset $C_k \subset \mathcal{L}$. Ideally, if all the incidents have been reported in the ITS, every one of these representative tickets is expected to represent a single incident in the network, *i.e.,* $G = \mathrm{I}$, where I is the total number of incidents.

$$\mathcal{G} = \{\bar{T}_1, \bar{T}_2, \ldots, \bar{T}_G\}$$

6 Experimental Results

In order to evaluate the proposed ticket correlation algorithms, datasets of actual ITSs are analyzed. It should be pointed out that it is difficult to access to this kind of information, as there are no ITSs data from companies publicly available. On one hand, vendors do not motivate the publication of these data due to competitive issues. On the other hand, network management teams have also the same lack of motivation for not publishing them, because this information is considered confidential and its publications might lead to security issues. We have only found some research and educational oriented networks that make their ITSs publicly accessible on the Internet [32].

In our case, the experimental data is composed of two different datasets, namely DS1 and DS2, that belong to the ITS of the same company, but are generated by two different staff members belonging to two different outsourcing companies. The

company that outsources the management of the ITS system is in charge of the management of a regional network formed by a large number of governmental sectors such as academic institutions, health centers, and service centers among others. For privacy reasons, and considering that we are working with sensitive issues regarding management efficiency, we keep some administrative information regarding the management companies hidden, such as the companies names and the regional network in charge of these companies.

6.1 Datasets Description

Dataset 1 (DS1) is composed of tickets for the whole year of 2011, with a total set $\mathcal{O}$ of $O = 19{,}162$ tickets. It is necessary to mention here that we have obtained DS1 from the first outsourcing company split into twelve batches, which are separated in time, each one for every month. On the other hand, Dataset 2 (DS2) is obtained from the second outsourcing company as one large batch file of 6 months, from October 2013 to the end of March 2014, with a total set $\mathcal{O}$ of $O = 9{,}612$ tickets.

Table 1 gives an overview of some useful statistics for both datasets. As shown in the table, there are three management groups that have the responsibility to manually create tickets, MS is the main technical group that creates tickets in response to receiving network alerts, while SD1 and SD2 are two different Service Desks (SD), where the staff creates tickets in response to receiving customer calls. SD1 is mainly referred to as call center level 1, whereas SD2 is referred to as call center level 2.

Our strategy for evaluating the correlation algorithms is as follows: for both datasets, we apply the correlation algorithms to extract the results and give some useful findings. We keep the data of DS1 split into twelve batches and repeat all of the experiments on the twelve batches separately, in order to study the behavior of the management team during the whole year. For the simplicity of the analysis, we ignore any border issues that may occur between any two batches that belong to two

Table 1 Some useful statistics of both datasets

	DS1		DS2
Total # of tickets	19,162		9,612
Mean # of tickets/month	1,596.5		1,602
Mean # of OIDs/ticket	1.4		1.42
Mean # of TIDs/ticket	1.2		1.39
Percentage of created tickets per management group	MS	45 %	54 %
	SD1	44 %	39 %
	SD2	11 %	7 %
Mean ticket lifetime [h:m:s]	22:48:02		33:41:37
Minimal ticket lifetime [h:m:s]	0:00:24		00:00:23
Maximal ticket lifetime [h:m:s]	1244:09:42		1397:13:27
Ticket intensity [tickets/hr]	2.22		2.19

consecutive months. Figure 5 shows the number of tickets created in each month for DS1.

An alignment of the fields of the tickets in both datasets is done with our proposed ticket model fields. It is presented in Table 2. In these datasets, the alignment has been done easily since each selected field clearly matches with a field in the proposed ticket model. This fact emphasizes the generality and simplicity of the proposed model, that consists of the most common fields that are found in any ITS.

6.2 Performance Indicators

In this section, we first suggest some performance metrics that help us to evaluate the efficiency of the proposed correlation algorithms.

The process of normalizing the metrics is not straightforward, as we are dealing with two types of tickets, *i.e.,* those extracted from the original database, and the representative tickets that are produced by the correlation process using the proposed correlation algorithms. For this reason, the estimation of the metrics will depend on each phase, where we have a different input database, *i.e.,* irrelevant tickets should be extracted from the original database during the preprocessing phase; locally correlated tickets should be extracted from the filtered database and replaced by their representatives; and finally, globally correlated tickets should also be extracted and replaced from the locally correlated database. In the following, we propose some metrics.

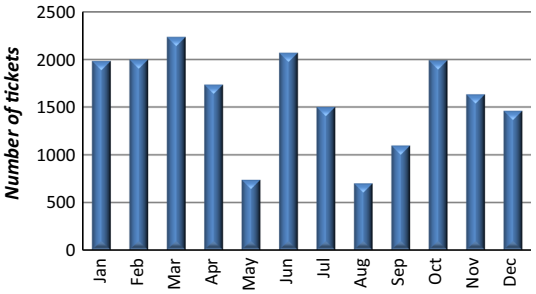

Fig. 5 Total number of tickets created in each month of DS1

Table 2 Alignment of considered ITS fields with incident ticket model fields

Company ITS fields	Proposed incident ticket model fields
CREATION_TIME	t_i^{CT}
RESOLUTION_TIME	t_i^{RT}
CLOSING_TIME	t_i^{VT}
LOCATION_ID	OID_i^1
CASE_ID	TID_i^1
ASSIGNED_TO_GROUP	TR_i^1

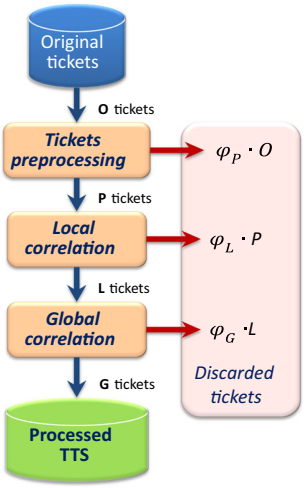

Fig. 6 Evolution of the number of tickets along the proposed system

The global process is depicted in Fig. 6. Let O be the number of records in the original database $\mathcal{O}$. Let P be the number of records in the filtered database, $\mathcal{P}$, *i.e.,* after removing irrelevant tickets. Let L be the number of records in the locally correlated database, $\mathcal{L}$, *i.e.,* after substituting each subset of locally correlated tickets for a local representative ticket. And finally, let G be the number of records in the globally correlated database, $\mathcal{G}$, *i.e.,* after substituting each subset of globally correlated tickets for a global representative ticket.

In every step, part of the tickets is removed from the input, as they are supposed to be useless for incident solving (preprocessing) or redundant (local and global correlation). In the latter case, although the original tickets are removed, a new representative ticket summarizing each subset of correlated tickets is introduced. Therefore, in order to assess each of the involved steps, we define a measure concerning the percentage of tickets that are removed in each phase.

The tickets reduction percentage, φ_x, for every correlation phase x, is defined as

$$\varphi_x = \left(\frac{input_x - output_x}{input_x}\right) \cdot 100, \quad \text{where } x \in \{\text{Preprocessing}(P), \text{Local}(L), \text{Global}(G)\} \tag{7}$$

where $input_x$ and $output_x$ are respectively the number of incoming/outgoing tickets for phase x. After applying all of the above correlation and preprocessing phases, our hypothesis is that the remaining number of tickets in the processed database, G, equals to approximately the number of incidents, I.

Starting from Eq. (7), we define the overall tickets' reduction percentage, $\varphi_{overall}$, by substituting $input_x$ with O and $output_x$ with G, *i.e.,*

$$\varphi_{overall} = \left(\frac{O - G}{O}\right) \cdot 100 \tag{8}$$

It should be noted that

$$(1 - \varphi_{overall}) = (1 - \varphi_P)(1 - \varphi_L)(1 - \varphi_G) \tag{9}$$

Also, we define the ticket creator's efficiency (E) for every phase x as:

$$E_x = 100 - \varphi_x, \quad \text{where } x \in \{P, L, G, overall\} \tag{10}$$

As said, the ideal ITS would be that in which the number of tickets generated equals the number of incidents, *i.e.,* I = O. In this case, according to Eq. (10), $\varphi_{overall} = 0$ %, that is, there are no redundant nor irrelevant tickets and, therefore, $E_{overall} = 100$ %.

6.3 Evaluating the Tickets Preprocessing Phase

We now use the above parameters as performance indicators to measure the reduction percentages and the overall efficiency of the tickets' creation process in both datasets. As indicated above, to avoid undesired border effects due to the structure of the provided datasets (lack of information at the beginning/end of the months in the provided information), we keep DS1 split into months and show the results for each one of them separately, besides the results obtained from DS2.

After applying the preprocessing criteria presented in Sect. 5.1 to extract *incident-related tickets*, Fig. 7 illustrates the reduction percentage of irrelevant tickets (φ_P) for each month of DS1 and DS2. On average, 11.4 % (DS1) and 15.6 % (DS2), respectively, of the original tickets are considered irrelevant from the point of view of incident solving. The final number of *incident-related tickets* after completing the preprocessing phase is P = 17,008 (DS1) and 8,105 (DS2), respectively.

Although many of the irrelevant tickets are really created by management staff, it is clear from the DS1 results that the own ITS had several malfunctions, especially during May, July and August, due to the high creation rate of such kind of tickets.

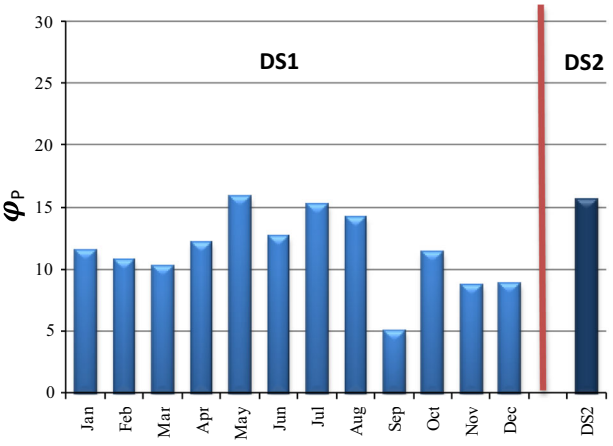

Fig. 7 Reduction percentage of irrelevant tickets extracted from both datasets

So, we conclude that the behavior of the management team regarding the creation of irrelevant tickets is non stable during the whole year. This methodology would point to a revision of the configuration of the platform in order to improve this performance.

We conclude that, in the case study, both the own ITS and the management staff always generate a large number of tickets that are not related to real network incidents. These types of tickets may have negative effects on the performance of the ITS as a whole, because creating such kind of tickets consume time and labor work, which are two important factors that affect the revenue and Quality of Service (QoS) of any management company.

6.4 Evaluating the Tickets Correlation Phase

Here, we evaluate both local and global correlation processes separately in order to see the effect of each of them on the whole system.

6.4.1 Evaluating the Local Correlation Algorithm

In order to extract the locally correlated tickets, we apply the local correlation algorithm to the filtered databases, which was discussed in Sect. 5.2.1. Our main assumption here is that two tickets are locally correlated if they share the same *main object ID* and they temporally overlap. We divide the evaluation phase into two subsections. First, we analyze the datasets and extract results. Second, we present two methods to validate the correlation results.

6.4.1.1 Results for the Local Correlation The reduction percentage of locally correlated tickets (φ_L), which is extracted for each month in DS1 and for the whole period in DS2 is shown in Fig. 8. Furthermore, Table 3 gives an overview of some useful statistics for three randomly selected months of DS1 and also for DS2: the number of processed tickets, P; the number of locally representative tickets, L; the

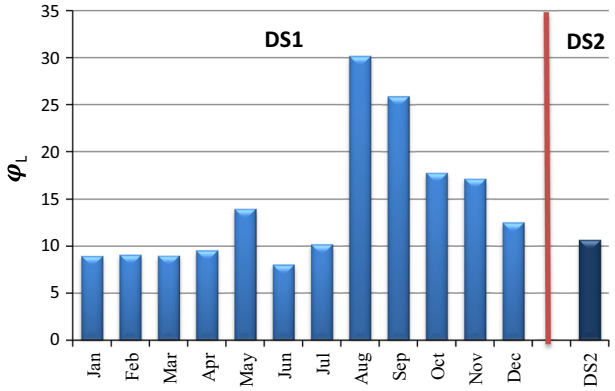

Fig. 8 Reduction percentage of locally correlated tickets extracted from both datasets

Table 3 Local correlation results for both datasets

Dataset	Period	# Input tickets (P)	# Output tickets (L)	φ_L	Locally correlated subsets			
					# Correlated tickets	# Representative subsets	#Tickets/subset	
							Mean	SD
DS1	Jan	1,756	1,598	9	270	112	2.41	0.83
	Jun	1,808	1,662	8.1	248	102	2.43	0.64
	Dec	1,331	1,164	12.5	308	141	2.18	0.82
DS2	6M	8,105	7,240	10.7	1,569	704	2.2	0.64

reduction percentage, φ_L; and some information related to the locally correlated subsets, *i.e.*, the number of tickets that have been correlated, the number of subsets of correlated tickets (# representative subsets) and the size of the subsets (mean and SD). From these results, we have the following findings: first, the proposed local correlation algorithm is able to discover, on average, more than 14.4 % (DS1) and 10.7 % (DS2) of the created tickets as redundant tickets. Second, we observe from the DS1 results that the behavior of the management team in creating locally correlated tickets during the whole year is somehow stable with few variations in the second half of the year, where about 21 % of the tickets are considered redundant.

6.4.1.2 Validation Process Since we are dealing with an unsupervised database, *i.e.,* it does not have tags indicating the incident associated to every ticket, there is an inherent difficulty in the validation process itself. Notwithstanding this, here we try to validate the usefulness of the proposed local correlation algorithm using two different validation methods. First, our hypothesis is in any ITS, if several tickets have the same *main object ID* (*node*, *service*, *location*, etc.), they overlap in time,

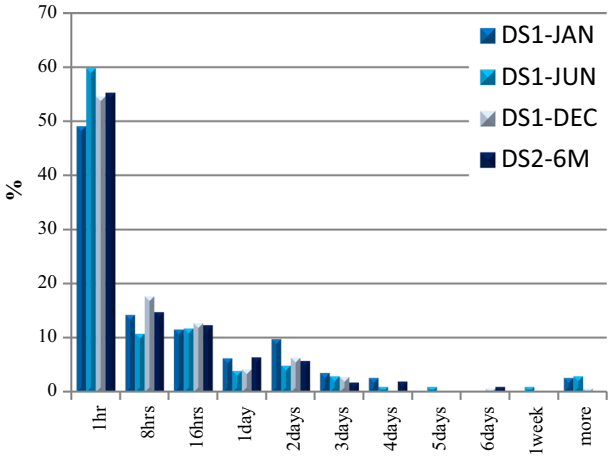

Fig. 9 Distribution of the intergroup delays of locally correlated tickets for subsets from both datasets

and they are resolved at almost the same time; therefore, we can assume that they really belong to the same incident. Hence, a study of the distribution of the delay of resolution times for each subset of locally correlated tickets could help to validate our correlation process.

We apply this analysis to our datasets. For the same three randomly selected months of DS1 (Jan, June, Dec) and for DS2, Fig. 9 illustrates the distribution of the delay between the representative ticket resolution time and the mean value of the resolution times for all the tickets in the subset, that is

$$t_k'^{RT} - \frac{1}{N_k}\sum_{i=1}^{N_k} t_i^{RT}, \quad \forall T_i \subset S_k$$

This is an indicator of the maximum dispersion of the resolution times for the correlated tickets. From this figure, we observe that the delay distribution of both datasets, the three selected months in DS1 and the whole period in DS2, is very similar. The graph is decreasing exponentially with an average of more than 54.5 % (DS1) of the subsets having all the related tickets resolved within 1 h between them; more than 14.3 % are within the next 8 h and so on. For DS2, 58 % of the tickets in a group are resolved with a difference lower than 1 h, while 14 % are resolved within an 8 h difference. We found that more than 80 % of the groups of correlated tickets (81.3, 86.3, and 89.4 %, respectively for DS1 and 88.9 % for DS2) contain tickets with differences in the resolution times lower than 1 day. This analysis applied on two different datasets (generated by staff following different management procedures) is a good indicator of the correct behavior of the proposed correlation algorithm.

An additional validation, besides the delay analysis described before, was made after randomly selecting 100 samples from the groups of tickets; 20 from each of the three selected months of DS1, and 40 from DS2. Figure 10 represents a histogram of the number of correlated tickets per subset for the selected samples. For each sample subset, we manually inspected all the tickets and checked whether it can be considered as a true correlation (TC) or not. We checked that most of the fields in the tickets that contain text-free information, especially those that characterize any incident such as incident description, worklog history, and solution description,

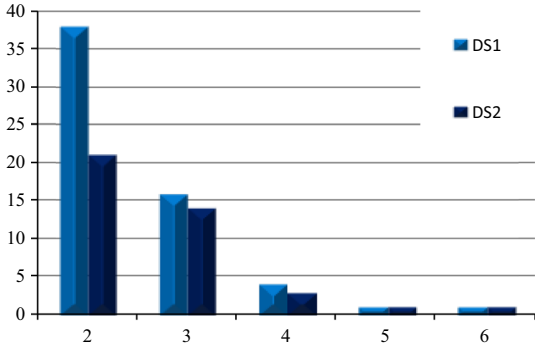

Fig. 10 Histogram of the number of tickets per subset for the selected samples from both datasets

share the same incident symptoms. Therefore, we argue that the efficiency of the local correlation algorithm is high, not finding any false positive in our sampling process. Thus, we conclude that our local correlation algorithm can correlate any subset of locally correlated tickets into a representative one that correctly describes the incident with a high confidence (efficiency is 100 % for the sampled subset). This finding emphasizes our claim that in any ITS, if several tickets have the same *main Object ID* and they temporally overlap, there is a high probability that they really belong to the same incident and should be correlated.

6.4.2 Evaluating the Global Correlation Algorithm

We apply the global correlation algorithm discussed in Sect. 5.2.2 to the locally correlated datasets, $\mathcal{L}$, *i.e.,* after every locally correlated subset is replaced by a representative ticket, in order to obtain the globally correlated tickets. We recall that we keep DS1 split into months and ignore the border issues that could happen between two consecutive batches. As in the local correlation phase, we divide the evaluation process into two subsections. First, we provide general results and extract several findings. Second, we use the same methods as in the previous subsection to validate the results.

6.4.2.1 Results for the Global Correlation Figure 11 shows the reduction percentage of globally correlated tickets (φ_G) that are extracted from the locally correlated dataset, $\mathcal{L}$. Furthermore, Table 4 gives an overview of some useful statistics of the same three selected months of DS1 and DS2 such as: the number of inputs to the global correlation process, L; the number of globally representative tickets, G; the reduction percentage, φ_G; and some information related to the globally correlated subsets, *i.e.*, the number of tickets that have been correlated, the number of subsets of correlated tickets (# representative subsets) and the size of the subsets (mean and SD). From these results we can extract some conclusions. First, we observe that there is another level of redundancy in which the proposed global

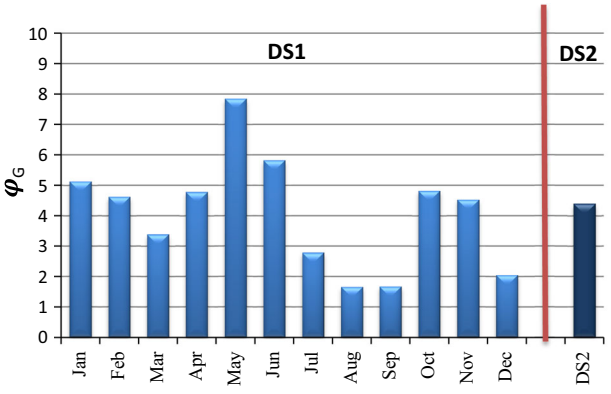

Fig. 11 Reduction percentage of globally correlated tickets extracted from both datasets

Table 4 Global correlation results for both datasets

Dataset	Period	# Input tickets (L)	# Output tickets (G)	φ_G	Global correlated subsets			
					# Correlated tickets	# Representative subsets	# Tickets/subset	
							Mean	SD
DS1	Jan	1,598	1,516	5.13	135	46	2.93	1.62
	Jun	1,662	1,565	5.84	165	60	2.75	1.82
	Dec	1,164	1,140	2.06	39	15	2.6	1.12
DS2	6M	7,240	6,922	4.4	516	198	2.6	2.42

correlation algorithm is able to discover, on average, about 4.1 % (DS1) and 4.4 % (DS2) of the created tickets as redundant tickets. Second, for DS1, unlike in the local case, there is a significant variance over the year, so it is less stable than the local correlation. Third, through a manual inspection of some samples of globally correlated tickets for both datasets we observe that management staff uses either *TID* or *OID* fields to relate tickets to each other and sometimes they use both in the same ticket.

The final number of tickets in the globally correlated database, G, is 14,181 (DS1) and 6,922 (DS2), which is also supposed to be the number of incidents, I.

6.4.2.2 Validation Process Here, we use the same validation methods as in the local correlation to validate the global correlation algorithm. Figure 12 illustrates the delay results. From this figure we observe that for DS1, on average, more than 86.5 % of the subsets have tickets with differences in resolution times lower than 1 day and for DS2, more than 88.1 %. Again, this finding is useful, since we have two datasets with different properties and that have coherent results of the global correlation. This confirms the validity of the proposed global correlation algorithm for us.

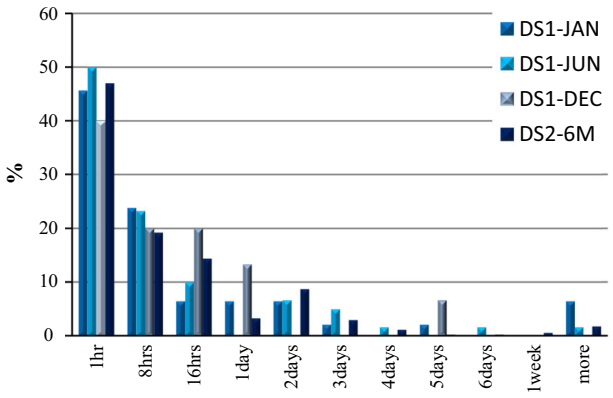

Fig. 12 Distribution of the intragroup delays for globally correlated tickets for subsets from both datasets

Second, for the sampling validation process, here we divide the analysis into two investigation periods, *delay <= 1 day* and *delay > 1 day*. Half of the samples are taken from the first investigation period, while the other half of the samples are taken from the second period. We show the validation results of 55 samples from DS1, which are also selected randomly except for December, since we have only 15 subsets in this month (we selected all of them), and 40 samples are taken from DS2 with a total of 95 samples. The histogram in Fig. 13 shows, for both datasets, the number of samples with the number of member tickets that they contain. The correlation results show that all of the samples taken from both datasets and belonging to the first investigation period (*delay <= 1 day*) are validated as TC, whereas five samples from the second investigation period, three from DS1 and two from DS2, (*delay > 1 day*) are validated as False Correlations (FC). Regarding the five samples validated as FC, we found out that the management staff sometimes refers to previous solved tickets if the ongoing incident has mainly the same preliminary symptoms. Sometimes, ticket creators relate a ticket with others just by writing *TID* or *OID* in some fields and do not describe the incident very well.

As a conclusion, the global correlation algorithm efficiency is really high (100 % of the sampled subset) in the first investigation period and 90 % (DS1)/95 % (DS2) in the second investigation period. On average, the global correlation algorithm efficiency for both datasets is 95 %.

6.4.3 Evaluating the Convergence of the Algorithms

We are also interested in evaluating the convergence of the algorithms when the provided datasets are split and the process is applied separately on the different subsets. Note that this is relevant, since the correlation process could be parallelized. Figure 14 illustrates the strategy that we follow to evaluate the stability. Here, we split DS2 into two parts: those tickets created by the main technical group (MS) and those created by the Service Desks SD1 and SD2. We apply the local correlation algorithm on each part separately (left part in Fig. 14), and obtain the results. Next, we again apply the local correlation algorithm to the already correlated tickets of

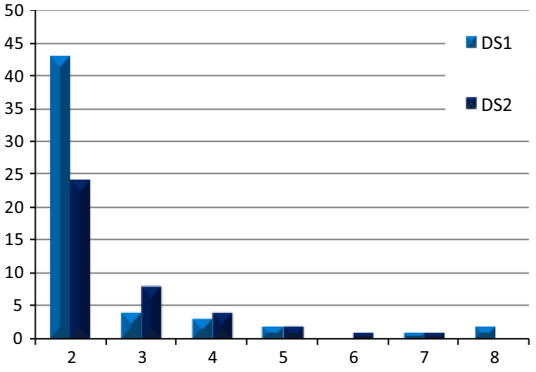

Fig. 13 Histogram of the number of tickets per subset for the selected samples from both datasets

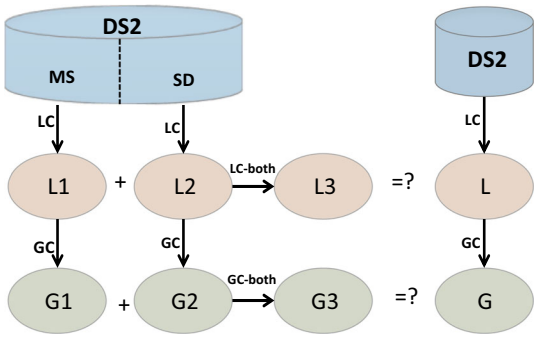

Fig. 14 Algorithms stability evaluation chart

both parts and compare the results with those extracted from the whole dataset (right part in Fig. 14). We do the same for the global correlation algorithm as well.

Table 5 shows the local/global correlation results of each part of the experiment. The number of local representative tickets of both datasets (L1 + L2) is 7,398, while the number of representative tickets for the output of the local correlation algorithm applied on L1 + L2 is 7,240, which is exactly the same result obtained in the previous experiments for the whole dataset.

Regarding the global correlation, the number of input tickets of both datasets (G1 + G2) is 7,135, and the output of the global correlation algorithm applied on this dataset (G3) obtains 6,922 representative tickets. We can check that this is also the same result presented in Sect. 6.4.2. Therefore, we conclude that, as expected, the results from local and global correlation algorithms are stable and independent on whether it is applied over a partitioned dataset.

7 Applications

Normally, and as we noted above, the process of handling the tickets by management staff is not completely systematic and may be incoherent. For example, the staff may create irrelevant and erroneous tickets for nonexistent network problems, or they may create redundant tickets, which may have different expressions or aspects of the same incident. On the other hand, there are a large number of commercial tools for ITSs, but most of them are focused on enhancing the whole system to recover the SLA in time. Regretfully, only few efforts have been devoted to improve the ticket generation process itself, which at the same time would improve the overall task of solving the incidents.

In this context, the proposed correlation method can be used not only to approximate the number of tickets to the number of incidents, but also as an assessment tool for measuring the quality of the ticket creation processes and teams. In this section, we present a case study in which we apply the proposed correlation method and metrics to the data described above in order to obtain some insights on the efficiency of the management staff and the processes they use, especially in the first stage of the management lifecycle, *i.e.,* the ticket creation stage. The results can

Table 5 Correlation results for local/global algorithms for the stability analysis

Correlation process	Dataset	# of Input tickets	# of Output tickets	Reduction percentage φ	Correlated subsets			
					# Correlated tickets	# Representative subsets	# Tickets/subset	
							Mean	SD
LC1	DS2-MS	4,929	4,456	$\varphi_L = 9.6$	907	434	2.1	0.37
	DS2-SD	3,176	2,942	$\varphi_L = 7.36$	402	168	2.4	0.95
LC-both	L1 + L2	7,398	7,240	$\varphi_L = 2.1$				
GC2	DS2-MS	4,456	4,310	$\varphi_G = 3.1$	259	113	2.29	1.07
	DS2-SD	2,942	2,825	$\varphi_G = 3.87$	174	57	3.05	3.8
GC-both	G1 + G2	7,135	6,922	$\varphi_G = 2.9$ %				

be useful from the point of view of improving both the management teams and the policies for ticket creation.

According to the proposed method, we measure the efficiency by obtaining the three levels of redundancy. For each level we calculate the reduction separately, as the actions to improve the handling of the tickets would be different at each level. As shown in Fig. 6 and explained across the previous sections, the whole process consists of three main steps. First, irrelevant tickets are filtered. Second, the number of tickets is reduced by substituting each subset of locally correlated tickets into a local representative ticket, using (for that purpose) the local correlation algorithm discussed in Sect. 5.2.1. Third, the total amount of tickets is decreased again using the global correlation algorithm proposed in Sect. 5.2.2 to obtain a global representative ticket for each subset of globally correlated tickets. The procedure takes the original ticket database ($\mathcal{O}$) as an input and provides the processed database ($\mathcal{G}$) as an output, that presents the different performance indicators for each step.

In the following subsections we take DS1 as a case study, and make all the necessary analyses and discussions based on it under the assumption that the final number of tickets (G) approximates the number of incidents (I).

7.1 Case Study: A Company ITS Assessment

The efficiency of the management staff for creating tickets can be obtained using Eq. (10) for each of the steps. The results obtained are shown in Fig. 15, where it is noticeable the high number of irrelevant tickets are up 16 %. From the point of view of the ITS irrelevant tickets should have not been created as they are not related to incidents and/or do not contain the minimum required information to be useful for incident solving. An inspection of those irrelevant tickets shows that many of them are not related to incident solving but other administrative issues, which can be considered a misuse of the ITS depending on the active policies. Once those

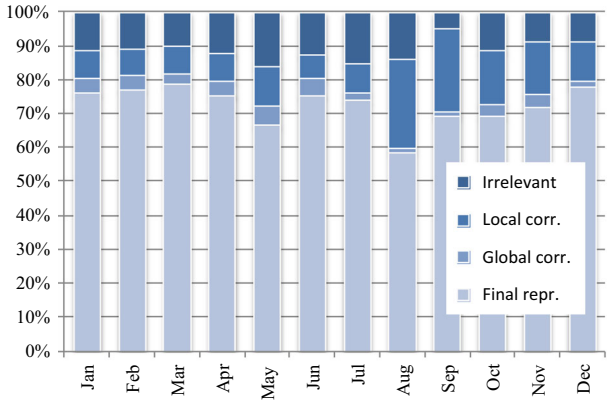

Fig. 15 Percentage of tickets removed in each step for DS1

irrelevant tickets are removed, the average efficiency for all the months is 72.6 %. As depicted in Fig. 15, a major redundancy in tickets can be found at the local level, which is not reasonable from the management's point of view as the related tickets are created for the same elements of the network.

If we analyze the results from a different point of view, a significant indicator would be the relationship between the number of incident related tickets (P) and the number of incidents (I). Assuming that G is approximately equal to I, Fig. 16 shows the number of tickets created for every incident. On average, there are 1.23 tickets/incident after filtering irrelevant tickets. This means that there is room for improvement in the creation and handling of the tickets.

In real situations, the three levels of redundancy usually have different effects on the overall efficiency of the company, *e.g.,* the cost of creating irrelevant tickets may not have the same weight as local or global redundant tickets. Therefore, instead of considering the overall efficiency—Eq. (10), the different partial efficiency measures can be used in a weighted form to evaluate the impact of each level of redundancy. This way, it would be possible to account for several reasons that play important roles, such as the degree of coordination between management groups, especially in a multi-team or multi-shift environment; updates on the calls and incidents coming in no matter who worked last on this particular incident; incidents extended over shifts; incidents that may be addressed by several different teams in the same shift; the mean time between correlated tickets, that may be used as a good indicator of the behavior of the staff for creating such kinds of redundant tickets and others like incident severity and SLA.

7.2 Insights from Evaluating Ticket Management Groups Procedures

As a complement to the assessment of the overall performance, it can be interesting to evaluate the amount of redundancy in the ticket creation process contributed by each of the management groups. Doing so might considerably help in identifying procedural problems and failures in coordination. Thus, in this subsection we study the behavior of each of the management groups in relation to the overall performance according to the proposed correlation model. As noted earlier, there are

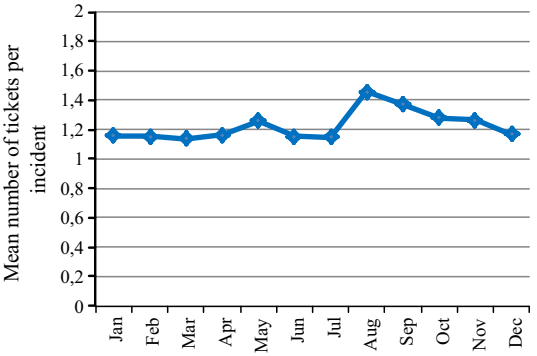

Fig. 16 Number of tickets created per incident

three groups which have the right to manually create tickets; they are MS, SD1, and SD2. The first one is the main technical group that creates tickets according to network alerts, while the last two are the Service Desks (SD), that create tickets in response to received customer calls.

7.2.1 Insights from the Preprocessing Analysis

The study of the distribution of irrelevant tickets generated by the three management groups reveals that only 1 % of them are created by MS, while up to 73 % are generated by SD1 and 26 % by SD2. Thus, we can conclude that SD1 presents very low efficiency in the ticket generation process due to the creation of a high amount of unnecessary tickets: more than three-quarters of these types of tickets were created by staff belonging to this group. These figures point to a potential fault in the procedures used by this group. Nevertheless, the nature of SD1 can partially explain this extremely high value compared to the others. As noted previously, SD1 is classified by the company as a call center level 1; that is, it is the first department that receives customers' calls. Consequently, the staff may create a large number of irrelevant tickets because they may receive many calls made by customers complaining about non-existent or non-networking related problems and/or providing insufficient information to properly identify the fault as a result of the customers not having enough knowledge about the normal operation of the system they are working with. Anyway, a review of the procedures used by this group is advisable. On the opposite side, the MS group created few irrelevant tickets, which is coherent with the procedures, as the staff belonging to this management group creates tickets based only on receiving alerts and thus, the chance of creating irrelevant tickets is very low.

7.2.2 Insights from the Local Correlation Analysis

Once those irrelevant tickets are filtered, the local correlation is used to identify those tickets that can be merged at this stage. Figure 17 illustrates the distribution of

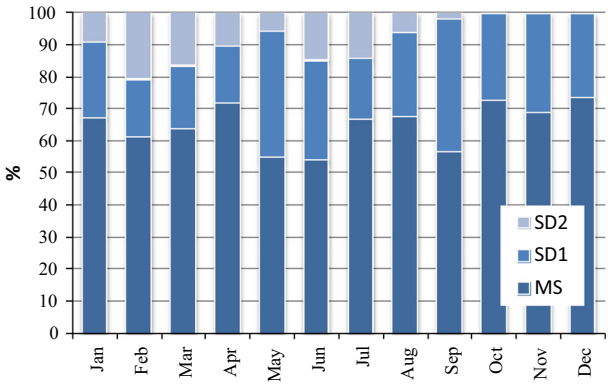

Fig. 17 Distribution of locally correlated tickets for the three management groups

those tickets according to the creator group. We observe that, on average, more than 65 % of the tickets are created by MS, 26.6 % by SD1, and finally 8.4 % by SD2. Unlike in the case of unnecessary tickets, a high percentage of these redundant tickets were created by the staff belonging to MS, *i.e.*, most of the redundancy is somehow related to MS.

In order to check whether the problem arises from tickets being created by more than one group or by duplicated tickets from the same group, an additional analysis is made. Thus, we take each group of locally correlated tickets, that is, those tickets that will be merged together by the local correlation, and examine whether they were created by the same or different management groups. As before, we apply this analysis over three randomly chosen months.

The results, shown in Fig. 18, proved that, on average, more than 73.5 % of the locally correlated subsets included tickets generated by a single group. This is a surprising result, as redundancy was expected to be mainly generated due to the existence of different management groups, each of them generating tickets for the same incident. A deeper insight into these results reveals that, from these subsets, 72.8 % of them are due to MS, 22.1 % to SD1, and 5.1 % to SD2. Obviously, the procedures used for ticket creation by MS should be revised. The existence of different working shifts can be at the origin of this redundancy: the staff belonging to MS is divided into three work shifts in a day, each lasting for 8 h. In our opinion, the lack of coordination procedures inside the same management group (intra-management) can explain the figures. For example, the staff belonging to the night shift may have created a ticket after receiving some alerts and on the next day, a person in the morning shift may have also created a duplicated ticket for the same incident, as some alerts are still appearing at the operator console. In this scenario, the local correlation procedure could be easily used in real time as a filter to detect these situations and avoid additional tickets.

Keeping on with the original analysis, 26.2 % of the locally correlated subsets include tickets created from two groups, while only 0.3 % of them where generated from the three groups. A manual inspection of many samples revealed that, as expected, the bulk of tickets coming from two different groups involved MS and SD1. Obviously, this is due to a lack of coordination between management groups (inter-management). For example, if a given element of the network is down, *i.e.*, a router, the staff at SD1 may receive calls from end users complaining about some

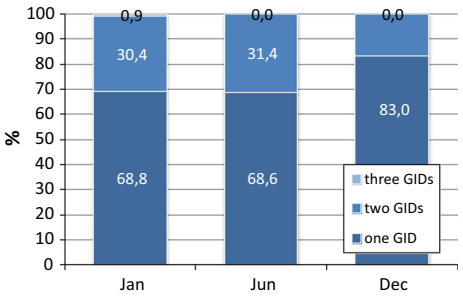

Fig. 18 Distribution of the number of locally correlated subsets as generated by management groups

problems in accessing services or applications affected by this element. Consequently, a ticket containing some preliminary information about the ongoing incident is created. At the same time, the staff at MS may receive alerts triggered by the same element announcing the existence of the same incident. Consequently, the staff creates another ticket related to the same incident. Therefore, this lack of coordination between different groups can lead to the creation of many redundant tickets.

7.2.3 Insights from the Global Correlation Analysis

Similarly to the local analysis, Fig. 19 illustrates the distribution of the globally correlated tickets for the three management groups. As in the local case, it is MS who is responsible for the majority of the tickets (64 %), followed by SD1 (30 %) and SD2 (6 %). The analysis of the sources for groups of related tickets is summarized in Fig. 20, with behavior similar to the case of local correlation. Thus, more than 70 % of the groups of related tickets contain tickets that were created by the same management group, again with MS being the dominant (up to 69.3 % of them).

A manual inspection of many samples of global correlated subsets reveals that staff at MS relate several tickets to each other if they are located nearly in the same network region. For example, the staff may receive many alerts triggered from two different network elements located nearly in the same geographical region. Consequently, they create two different tickets related to the same incident.

The remaining subsets contain tickets created by two different GIDs, namely MS and SD1. As before, this can be due to the lack of coordination between the management groups involved in incident resolving tasks.

8 Conclusions and Future Work

In this paper, we propose a novel, simple, and effective approach to correlate and merge incident tickets in an ITS. The approach is based on a generic model for the

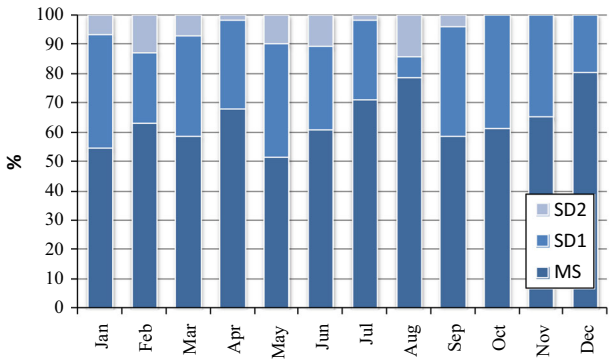

Fig. 19 Distribution of globally correlated tickets for the three management groups

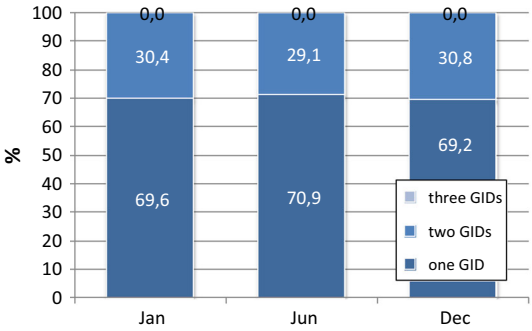

Fig. 20 Distribution of the number of globally correlated subsets as generated by management groups

tickets that preserves and categorizes the relevant information and enables the comparison of their properties with relatively simple functions. No additional sources of information, apart from the tickets themselves and the information they contain, is required during the merging procedure. Despite its simplicity, the model has revealed to be useful to reduce the number of tickets generated and handled by ITS users, which is a desirable target in order to improve the workflow in a management company.

The experiments on two different datasets from a real company have shown that contrary to what is expected, there are significant amount of redundant tickets being generated by the different actors in the ITS. The proposed model and method can be easily incorporated in the in-production system, not allowing the generation of additional tickets when a related one is active, or automatically removing those detected as redundant. The first is quite straightforward for the local level of the algorithm, which can be applied in real-time at the time of the creation of the ticket. The global phase uses information that could not be available at the time of the creation of the ticket but added later and, therefore, would imply an a posteriori filtering of the tickets.

As an additional use of the model and the proposed procedure, we have also described through a case study how it can be used as an assessment tool to provide some measures about the management staff's efficiency at the ticket creation process. This way, it is possible to identify some deficiencies in the procedures, policies, or behaviors of the different actors involved in the ITS management process, enabling corrective actions to be taken and evaluated.

Part of the potential of the solution can be attributed to the different nature of the information provided by some of the actors, *i.e.*, information gathered from automatically generated alerts *vs*. information from customers, which can be complementary. Somehow, this is including some semantic information in the process of correlating tickets. An extension of this idea is currently being developed and tested by the authors to the alert correlation problem by including information from the tickets through the proposed model into the process. This way, an alert-and-tickets correlation procedure is under study. We are confident that these models will help to reduce the number of alerts (to be handled) to a minimum.

Acknowledgments This work has been partially supported by Spanish MICINN through Project TEC2011-22579.

References

1. Marilly, E., Martinot, O., Papini, H., Goderis D.: Service level agreements: a main challenge for next generation networks. In: Proceedings of the 2nd European Conference on Universal Multiservice Networks (ECUMN), pp. 297–304 (2002)
2. Vasseur, J.P., Pickavet, M., Demeester, P.: Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS. Morgan Kaufmann Publishers Inc., San Francisco (2004)
3. Turner, D., Levchenko, K., Snoeren, A., Savage, S.: California fault lines: understanding the causes and impact of network failures. In: Proceedings of the ACM SIGCOMM Computer Communication Conference (SIGCOMM '10), pp. 315–326 (2010)
4. Gill, P., Jain, J., Nagappan, N.: Understanding network failures in data centers: measurement, analysis, and implications. In: Proceedings of the ACM SIGCOMM Computer Communication Conference (SIGCOMM '11), pp. 350–361(2011)
5. IT Infrastructure Library, Office of Government Commerce (UK). http://www.itil-officialsite.com/. Accessed 26 June 2014
6. OGC (Office of Government Commerce) Ed., Service Operation, ser. IT Infrastructure Library v3 (ITIL v3). The Stationary Office, Norwich (2007)
7. OpenView, HP. http://www8.hp.com/us/en/software/enterprise-software.html. Accessed 26 June 2014
8. Lewis, L., Dreo, G.: Extending trouble ticket systems to fault diagnostics. IEEE Netw. **7**(6), 44–51 (1993)
9. Johnson, D.: NOC Internal Integrated Trouble Ticket System Functional Specification Wishset. RFC 1297 (1992)
10. Dreo, G.: A framework for supporting fault diagnosis in integrated network and systems management: methodologies for the correlation of trouble tickets and access to problem–solving expertise. Ph.D. Dissertation, Ludwig–Maximilians–Universität München (1995)
11. Shao, Q., Chen, Y., Tao, S., Yan, X., Anerousis, N.: Efficient ticket routing by resolution sequence mining. In: Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 605–613 (2008)
12. Miao, G., Moser, L., Yan, X., Tao, S., Chen, Y., Anerousis, N.: Generative models for ticket resolution in expert networks. In: Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 733–742 (2010)
13. Tang, L., Li, T., Shwartz, L., Grabarnik, G.Y.: Identifying missed monitoring alerts based on unstructured incident tickets. In: Proceedings of the 9th International Conference on Network and Service Management (CNSM), pp. 14–18 (2013)
14. Li, Y., Li, T.H.: A method of effort estimation for incident tickets in IT services. In: Proceedings of the IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), pp. 311–316 (2013)
15. Marcu, P., Grabarnik, G., Luan, L., Rosu, D., Shwartz, L., Ward, C.: Towards an optimized model of incident ticket correlation. In: Proceedings of the IFIP/IEEE International Symposium of Integrated Network Management, (IM), pp. 569–576 (2009)
16. Hanemann, A.: Automated IT service fault management based on event correlation techniques. Ph.D. Dissertation, University of Munich, Department of Computer Science, Munich (2007)
17. Hanemann, A., Marcu, P.: Algorithm design and application of service oriented event correlation. In: Proceedings of the 3rd IFIP/IEEE International Workshop on Business Driven IT Management (BDIM), pp. 61–70 (2008)
18. Tang, L., Li, T., Shwartz, L., Pinel, F., Grabarnik, G.: An integrated framework for optimizing automatic monitoring systems in large IT infrastructures. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 1249–1257 (2013)
19. Medem, A, Teixeira, R., Feamster, N., Meulle, M.: Joint analysis of network incidents and intradomain routing changes. In: Proceedings of the International Conference on Network and Service Management (CNSM), pp. 198–205 (2010)

20. Medem, A., Teixeira, R., Feamster, N., Meulle, M.: Determining the causes of intradomain routing changes. Technical Report. UMIACS University (2009)
21. Feamster, N., Balakrishnan, H.: Detecting BGP configuration faults with static analysis. In: Proceedings of the USENIX 2nd International Conference on Symposium on Networked Systems Design and Implementation (NSDI), pp. 43–56 (2005)
22. Turner, D., Levchenko, K., Mogul, J.C, Savage, S., Snoeren, A.C.: On failure in managed enterprise networks. HP Labs HPL-2012-101 (2012)
23. Potharaju, R., Jain, N., Nita-Rotaru, C.: Juggling the jigsaw: towards automated problem inference from network trouble tickets. In: Proceedings of the USENIX 10th Symposium on Networked Systems Design and Implementation (NSDI), pp. 127–141 (2013)
24. Medem, A., Akodjenou, I., Teixeira, R.: TroubleMiner: mining network trouble tickets. In: Proceedings of the IFIP/IEEE International Conference on Integrated Network Management-Workshops, (IM), pp. 113–119 (2009)
25. Tanaka, J.: Analysis of trouble tickets issued by APAN JP NOC. www2.jp.apan.net/meetings/busan03/noc/tanaka.ppt (2003). Accessed 26 June 2014
26. Pándi, Z.: Analysis of public trouble ticket data. Technical Report. Budapest University of Technology and Economics, Department of Telecommunications (2005)
27. Huang, Y., Feamster, N., Lakhina, A., Xu, J.: Diagnosing network disruptions with network-wide analysis. In: Proceedings of the International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), pp. 61–72 (2007)
28. Labovitz, C., Ahuja, A., Farnam, J.: Experimental study of internet stability and backbone failures. In: Proceedings of the 29th International Symposium of Fault-Tolerant Computing, pp. 278–285 (1999)
29. ServiceNow, Inc. http://www.servicenow.com/products/it-service-automation-applications/incident-management.html. Accessed 26 June 2014
30. Remedy, BMC. http://www.bmc.com/it-solutions/remedy-itsm.html/. Accessed 26 June 2014
31. Tivoli, IBM. http://www-01.ibm.com/software/tivoli/. Accessed 26 June 2014
32. http://www.switch.ch/network/operation/tts/. Accessed 26 June 2014

Saeed Salah received the B.Sc. degree in Electrical Engineering from Al-Najah National University in 2003 and M.Sc. degree in Computer Science from Al-Quds University in 2009. He then worked at the same University as an instructor at the Department of Computer Science. He is currently a Ph.D. student at the Department of Signal Theory, Telematics and Communications of the University of Granada (Spain). His research interests include the network management and configurations, event correlation, network architecture, network security and routing protocols.

Gabriel Maciá-Fernández is an Associate Professor at the Department of Signal Theory, Telematics and Communications of the University of Granada (Spain). He received a MS in Telecommunications Engineering from the University of Seville, Spain, and the Ph.D. in Telecommunications Engineering from the University of Granada. In the period 1999–2005, he worked as a specialist consultant at "Vodafone España". His research interests are focused on computer and network security, with special focus on intrusion detection, reliable protocol design, network information leakage and denial of service.

Jesús E. Díaz-Verdejo is Professor in the Department of Signal Theory, Telematics and Communications of the University of Granada (Spain). He received his B.Sc. in Physics (Electronics speciality) from the University of Granada in 1989 and his Ph.D. in Physics in 1995. Since 1990 he is a lecturer at this University. His initial research interest was related with speech technologies. He is currently working in computer networks, mainly in computer and network security and traffic identification.

Leovigildo Sánchez-Casado received his M.Sc. degree in Telecommunications Engineering in 2008, his M.Sc. degree in Electronics Engineering in 2009 and his Ph.D. in the field of Networking/Telematics in 2014, each from the University of Granada (Spain). In 2010, he joined the Department of Signal Theory, Telematics and Communications of the University of Granada as a researcher. He is a member of the "Network Engineering and Security Group (NESG)". His research interests are mainly focused on the area of network security and more specifically on intrusion detection and response, principally in ad hoc networks.